웹에 기반한 개방형 분산 HW/SW 통합설계 환경

김승권[†] · 김종훈[††]

요 약

하드웨어와 소프트웨어로 구성된 시스템을 동시협조 설계하는 새로운 설계 패러다임인 HW/SW 통합설계를 지원하는 설계 도구가 많이 개발되고 있다. 기존의 HW/SW 통합설계 도구는 빠른 기술 변화, 제한된 플랫폼, 표준화되지 않은 시스템 기술, 일관적이지 못한 사용자 인터페이스, 개발 대상의 다양성, 지원 기능의 차이 등으로 인해 일반화되지 못하고 있다. 본 논문에서는 이런 문제점을 해결하기 위해, 웹에 기반한 개방형 분산 HW/SW 통합설계 환경을 제안한다. 제안한 HW/SW 통합설계 환경은 객체에 기반한 3층 클라이언트/서버 구조를 가지며, 특정 벤더에 구애받지 않는 개방성을 지닌다. 또한 세션 서비스를 이용한 협동 작업 환경을 지원하며, OOUI를 채택해 사용자 인터페이스를 크게 변경하지 않고 새로운 기능을 쉽게 추가할 수 있다. 제안한 환경은 효율적이고 안전한 설계 데이터의 전송을 보장하기 위해 트랜잭션 서버와 보안 서비스를 포함한다. 제안한 환경의 타당성을 입증하기 위해 웹에 기반한 개방형 분산 HW/SW 통합설계 환경의 프로토타입인 WebCEDA를 개발하였다. 제안한 환경은 일반적인 CAD 도구의 웹-기반 환경의 모델로 활용될 수 있다.

Web-based Open Distributed HW/SW Codesign Environment

Seung-Kwon Kim[†] and Jong-Hoon Kim[††]

ABSTRACT

HW/SW codesign is integrated design of systems implemented using both hardware and software components. Many design tools has been developed to support this new paradigm, so far. Current codesign tools are not widely used as been expected because of variety problems - rapidly evolving technology, platform dependency, absence of standard specification method, inconsistent user interface, varying target system, different functionality. In this paper, we propose a web-based distributed HW/SW codesign environment to remedy this kinds of problem. Our codesign environment has object-based 3 tier client/server architecture. It supports collaborative workspace through session service. Fully object-oriented design of user interface(OOUI) enables easy extension without change of user interface. Furthermore it contains transaction server and security server for efficient and safe transfer of design data. To show a validity of our design, we developed prototype of web-based HW/SW codesign environment called WebCEDA. Our model of HW/SW codesign can be used for web-based generic CAD tools.

1. 서 론

HW/SW 통합설계란 하드웨어와 소프트웨어로 구성된 시스템을 동시협조 설계하는 통합된 설계 패러다임을 말한다[1]. VLSI/CAD 기술의 발달로 디지털 시스템의 복잡도가 증가하고 시스템의 출시 기간이 점점 단축되면서 하드웨어와 소프트웨어의 통합적인 설계를 위한 설계 방법론에 대한 관심이 증대되고 있다. HW/SW 통합설계는 설계 절차를 가속화하여 제품의 출시 기간을 줄여주며, 하드웨어와 소프트웨어의 구현 비율을 동적으로 조절(Trade-off)할 수 있어서 성능상의 제약 조건을 만족하는 작은 비용의

이 논문은 1997년도 한국학술진흥재단의 학술연구조성비에 의하여 지원되었음.

† 동아대학교 컴퓨터공학과
†† 동아대학교 컴퓨터공학과

제품을 제작하는 최적 설계를 가능하게 하는 장점이 있다. HW/SW 통합설계 절차는 일반적으로 ① 요구사항 분석, ② 시스템 기술(Specification), ③ HW/SW 분할(Partitioning), ④ 인터페이스 생성, ⑤ 하드웨어/소프트웨어 합성, ⑥ 통합시뮬레이션(Cosimulation), ⑦ 시스템 결합 및 성능 평가의 과정으로 이루어진다. 현재 통합설계 기법은 명령어 집합 프로세서, ASIP(Application Specific Instruction Processor), 내장형 시스템 등의 개발에 적용되고 있다. 전통적인 통합설계 절차는 초기 바인딩과 초기 설계 결정을 지원하는 단순한 방식이었지만, 그간의 연구 성과에 힘입어 모델-기반 방식으로 점차적으로 전환되고 있다. 모델-기반 통합설계 방식은 개선(Refinement) 과정을 추가하여 분할과 라이브러리 바인딩을 뒤로 미루어 설계 변화에 쉽게 대처할 수 있으며, 모듈화되고 계층적인 내부 모델의 사용으로 인해 컴포넌트를 재사용하는 장점이 있다. 통합설계에서 시스템 기술에 사용되는 언어는 VHDL, C와 같은 기존의 실행 가능한 시스템 기술 언어와, Ptolemy[7]와 같은 이질적(Heterogeneous) 기술 언어, Lotos[6], Esterel[5], SpecCharts 등과 같이 새롭게 개발된 언어 등이며, FSM, DFG, CDFG등이 내부 모델로 사용되고 있다.

현재 발표된 HW/SW 통합설계 도구는 Polis [9], CoDe-X, COOL[10], PeaCE[11], Chinook[12], CodeSign[8], SpecSyn[13] 등이 있다. 기존의 통합설계 도구는 개발 대상의 차이와 HW/SW 통합설계의 표준화되지 않은 개발 절차로 인해 각기 다른 다양한 방법을 지원하며, 독자적인 사용자 인터페이스와 내부 모델을 가진다. 설계 절차를 보다 생산성있게 하기 위해서는 다양한 환경의 시스템 설계자들이 모두 접근할 수 있는 공통 인터페이스가 필요하며, 내부 모델의 변경과 같은 내부적인 기술 변화를 감출 수 있어야 한다. 또한 팀 단위의 설계와 자원 공유에 대한 수단이 고려되어야 한다. 최근 CAD/EDA 분야에서도 이런 문제점을 인식하여 Weld와 같은 통합된 웹-기반 환경의 개발에 착수하였다[2,3]. HW/SW 통합설계 도구는 개발 방법론의 다양성 때문에 내부 절차를 추상화할 필요가 있으며, 빠른 기술 발전으로 인해 생명 주기가 짧아 효과적으로 도구를 갱신할 수 있는 방법이 절실히 요구된다. 본 논문에서는 이런 문제점을 해결하기 위한 방안으로 웹에 기반한 개방형 분산 HW/SW 통합설계 환경을 제안하고, 제안한 환경의 타당성을 입증하기 위한 프로토타입으로 WebCEDA를 개발한다. 제안한 환경은 지리적으로 분산된 설계자에게 효율적으로 설계 작업을 수행할 수 있는 공동 작업장을 제공한다. 또한 향상된 기술을 사용자 인터페이스의 변화를 최소화하면서 빠른 시간 내에 제안한 통합설계 환경에 적용할 수 있다. 시스템 설계자는 상위 수준에서 문제 자체에 초점을 맞추어 시스템을 설계할 수 있으며, 제안한 환경의 개방성으로 인해 시스템 설계자가 사용하고 있는 기존 환경을 이용할 수 있기 때문에, 추가적인 시스템 구입 등의 비용을 절감할 수 있다.

본 논문에서 제안한 웹에 기반한 개방형 분산 HW/SW 통합설계 환경은 일반적인 CAD/EDA 분야의 분산 환경 구축을 위한 모델로서 활용될 수 있으며, 기존의 HW/SW 통합설계 도구를 이용하여 쉽게 기능을 확장할 수 있다.

본 논문은 다음과 같이 구성되어 있다. 먼저 2장에서 기존의 HW/SW 통합설계 도구를 분석하고 기존 도구의 문제점을 살펴본다. 그리고 새로 개발되는 HW/SW 통합설계 도구가 가져야 하는 요구 사항을 도출한다. 3장에서는 본 논문에서 제안하는 웹에 기반한 개방형 분산 HW/SW 통합설계 환경에 대해 기술한다. 4장에서는 제안한 환경의 프로토타입으로 WebCEDA를 개발한다. 5장에서는 이 연구를 요약하고 앞으로의 연구 방향을 제시한다.

2. 기존의 HW/SW 통합설계 도구의 문제점

HW/SW 통합설계를 지원하는 도구의 개발은 지금까지 많이 진행되어 왔지만, 그 중에 가장 널리 알려져 많이 사용되고 있는 도구는 Polis, CoDe-X, COOL, PeaCE, Chinook, CodeSign, SpecSyn 등이 있다. Polis는 내장형 시스템의 정형적 기술과 검증, 통합 시뮬레이션에 중점을 두고 개발한 환경으로, 소프트웨어 합성과 추정의 작업에 강점을 지니지만 분할 알고리즘이나 스케줄링에 대한 작업은 설계자의 몫으로 돌리고 있다. CoDe-X는 재구성가능한 데이터패쓰 하드웨어에 기반한 범용 가속기인 Xputer를 위한 2단계 HW/SW 통합설계 프레임웍으로, C를 시스템 기술 언어로 채택한 소프트웨어 지향적인 분할 도구를 가지고 있다. COOL은 데이터플로우 기반 시스템을 위해 개발된 HW/SW 통합설계 도구로,

VHDL의 일부를 시스템 기술 언어로 사용하는 하드웨어 지향적인 방법론을 사용한다. PeaCE는 이질적(Heterogeneous) 디지털 시스템의 빠른 개발을 위해 Ptolemy를 확장한 HW/SW 통합설계 환경으로, DFG를 시스템 기술시 이용하며 최근에는 DFG만으로는 한계를 느껴 확장된 FSM을 시스템 기술 방법의 하나로 추가함으로써, Ptolemy처럼 이질적 시스템 기술 언어를 사용한다. Chinook은 내장형 시스템을 위한 HW/SW 합성 도구로써, 분산 구조를 가지고, 시간 제약이 있는 제어 기반의 재반응(Reactive) 시스템의 설계를 위해 사용된다. 실시간 지원에 관한 연구를 추가하여 분산 내장형 시스템을 위한 ipChinook이 개발 중에 있다. CodeSign은 복잡한 실시간 내장형 시스템의 설계를 위한 통합설계 도구로써, Ptolemy와 같은 이질적 혼합 모델로 시스템 기술한다. SpecSyn은 내장형 시스템의 설계에 강점을 갖는 SpecCharts [4]를 시스템 기술 언어로 채택한 통합설계 도구이다. 표 1은 기존의 통합설계 도구를 구현 대상, 시스템 기술 언어, 내부 모델, 사용 환경을 기준으로 정리한 것이다.

표 1. 기존의 HW/SW 통합설계 도구

툴의 종류	구현 대상	시스템 기술 언어	내부 모델	사용 환경
Polis	내장형 시스템	Esterel	확장 FSM	단일 플랫폼
CoDe-X	Xputer	C	Flow Graph	단일 플랫폼
COOL	데이터 플로우 기반 시스템	VHDL 부분집합	State Transition Graph	단일 플랫폼
PeaCE	이질적 디지털 시스템	DFG + 확장 FSM	DFG + 확장 FSM	단일 플랫폼
Chinook	내장형 시스템	ACT 구성 모델	ACT 구성 모델	다중 플랫폼
CodeSign	복잡한 실시간 내장형 시스템	이질적 혼합 모델	FunState	다중 플랫폼
SpecSyn	내장형 시스템	SpecCharts	CDFG	단일 플랫폼

각 HW/SW 통합설계 도구는 구현하고자 하는 대상 시스템의 차이로 인해 서로 다른 시스템 기술 언어와 내부 모델을 사용하고 있으며, 빠른 기술 발전과 사용자 요구에 의해 계속 변경되고 있다. 각 도구에서 지원하는 기능도 다양하다. 모든 통합설계 절차를 지원하는 도구도 있고, Polis처럼 특정 기술만을 지원하고 나머지는 설계자에게 맡기는 경우도 있다. 또한 통합설계 도구의 기능은 빠른 속도로 업그레이드되고 있어, 생명 주기가 매우 짧다. 각 도구의 사용 플랫폼은 대부분이 스팍 머신과 Unix 운영체제로 제한되어 있으며, CodeSign과 Chinook의 최근 버전인 ipChinook의 경우만이 자바 기술을 이용하여 다중 플랫폼을 지원하지만 각각 설치 방법과 필요한 소프트웨어 구성이 다르다. 또한 기존의 통합설계 도구는 시스템 설계자의 협동 작업에 대한 고려가 전혀 되지 않고 있다. 기존의 HW/SW 통합설계 도구가 갖는 문제점을 요약하면 다음과 같다.

- 기존의 통합설계 도구는 HW/SW 통합설계를 연구하기 위한 실험적 환경이 대부분이고, 통합설계 기술의 발전 속도가 빨라서 기반 기술이 자주 변경된다. 시스템 설계자가 발전된 통합설계 기법을 이용하기 위해서는 새 버전의 통합설계 도구를 다운로드 받아서 다시 설치해야 하는 부담감이 있기 때문에, 발전된 기술을 사용자가 실제로 사용하기까지는 오랜 시간이 걸린다.
- 기존의 통합설계 도구는 대부분 단일 플랫폼으로 구성되어 있어서, 그 환경에 익숙하지 않은 시스템 설계자는 사용하기 어렵다. 또한 다중 플랫폼을 지원하는 Chinook과 CodeSign의 경우에도 각 플랫폼마다 설치 방법 및 필요한 구성 요소가 달라서 설치도 어려울 뿐 아니라 유지 보수에도 많은 비용이 든다.
- 기존의 통합설계 도구는 시스템 설계자의 공동 작업에 대한 배려가 없다.

이런 문제점을 해결하기 위해서는 개방성, 보안성, 공동 작업 환경, 기술 추가의 용이성, 쉬운 인터페이스를 갖춘 새로운 HW/SW 통합설계 도구의 개발이 필요하다.

3. 웹에 기반한 개방형 분산 HW/SW 통합설계 환경

기존의 HW/SW 통합설계 도구는 단일 컴퓨터 상

에서 수행되며 플랫폼에 제한적이다. 이것은 배포와 설치 및 사용을 어렵게 한다. 또한 빠른 기술 발전으로 인한 잦은 기술 변화와 적용 기술의 다양성은 유지 보수와 HW/SW 통합설계 도구의 저변 확대를 어렵게 하는 원인이 되고 있다. 본 연구에서는 이런 기존 도구의 문제점을 해결하기 위해 웹-기반 개방형 분산 HW/SW 통합설계 환경을 제안한다.

3.1 3층 클라이언트/서버 구조

그림 1은 제안한 환경이 가지는 3층 클라이언트/서버 구조를 보여준다. 2층 클라이언트/서버 환경은 단순성으로 인해 신속한 응용 프로그램의 개발이 가능하지만, HW/SW 통합설계와 같은 대규모 프로젝트에는 적합하지 않다. 3층 구조는 확장성과 유연성, 견고성이 요구되는 HW/SW 통합설계 분야에 적합하다. 제안한 환경이 3층 구조를 가지는 근거는 다음과 같다.

- 많은 응용 프로그램 서비스를 제공하며 50개 이상의 클래스가 구현에 사용된다.
- 응용 프로그램은 C++, Java, Perl 등과 같이 서로 다른 언어로 프로그래밍된다.
- 2개 이상의 이질적 데이터 소스가 사용된다.
- 응용 프로그램이 빠른 기술 변화로 인해 많은 변경과 추가가 예상된다.
- 시스템 설계시 많은 트랜잭션이 발생하며, 공동 작업시 동일 데이터베이스를 액세스하는 경우가 많다.

제안한 환경을 세분화하면 클라이언트와 HW/SW 통합설계 서버, 리모트 서버로 나눌 수 있다. 클라이언트는 사용자 인터페이스를 담당하는 부분으로 동적 웹 페이지로 구성된 사용자 인터페이스와 자바빈/Active X/CORBA 객체로 구성되어 설계 절차 관리, 시스템 기술, 시각화(Visualization) 등을 담당하는 클라이언트 프로그램이다. HW/SW 통합설계 서버는 응용 프로그램 논리(Application Logic)를 담당하는 부분으로, 웹 서버, 트랜잭션 서버, 응용 서버, 보안 서버로 구성된다. 리모트 서버는 백엔드 자원 관리(Backend Resource Manage)를 담당하는 부분으로 HTML 저장소, 문서 저장소, 파일 시스템, 데이터베이스, 통합설계에 사용되는 응용 프로그램으로 구성된다.

그림 1. 웹에 기반한 개방형 분산 HW/SW 통합설계 환경의 구조

제안한 구조는 응용 프로그램이 중앙 서버에서 관리됨으로 시스템 관리의 복잡성을 줄였다. 또한 데이터 수준의 보안이 아닌 서비스, 메서드, 개체 수준에서 보안 사항들을 세부 조정함으로써 보안성을 높였다. 여러 서버들로 로드를 분산시켜 성능과 확장이 뛰어나며, 애플릿이나 자바 빈과 같은 다운로드가 쉬운 얇은 클라이언트로 인터넷 지원이 뛰어나다. 또한 3층 구조를 채택함으로써 RPC형 호출 외에도 연결이 없는 메시징, 큐 삽입 전달, 게시-가입, 방송 등의 다양한 통신 옵션을 지원할 수 있다. 3계층 모두 다른 컴퓨터를 사용할 수도 있지만, 2번째와 3번째 층이 동일 컴퓨터 내에 존재할 수 있어 하드웨어 아키텍처의 유연성을 제공한다. 제안한 구조의 단점은 2층에 비해 상대적으로 겪게 되는 개발상의 어려움이다. 그러나, 최근 개발되고 있는 CORBA 등의 표준 도구로 인해 이런 개발상의 어려움은 점점 줄어들고 있다.

3.2 개선된 HW/SW 통합설계 환경

제안한 웹-기반 개방형 분산 HW/SW 통합설계 환경은 기존의 HW/SW 통합설계 도구의 문제점을 세션 서비스를 이용한 협동 작업 환경의 제공, OOUI를 이용한 사용자 인터페이스 변경의 최소화, 트랜잭션 관리를 통한 효율적인 설계 데이터의 전송, 보안 서버를 통한 안정성 확보 등의 기능으로 해결한다.

(1) 세션 서비스를 이용한 협동 작업 환경

HW/SW 통합설계 분야는 구현하고자 하는 대상

시스템의 복잡도가 높아서 많은 시스템 설계자가 하나의 프로젝트에 참여하는 경우가 대부분이다. 이 경우 시스템 설계자의 정보 교환과 지리적으로 떨어져 있는 설계자가 함께 일할 수 있는 협동 작업 환경에 대한 고려가 반드시 있어야 함에도 불구하고, 기존의 HW/SW 통합설계 도구는 이에 대한 배려를 전혀 하지 못했다. 효율적이고 신뢰할 수 있는 협동 작업 환경(Collaborative Environment)을 제공하기 위해서는 플랫폼에 독립적이고 보안성이 제공되어야 한다. 또한 사용자가 협동 작업을 생성, 삭제, 참여, 탈퇴할 수 있는 기능도 제공되어야 한다[14]. 세션 서비스를 추가하면 이런 협동 작업 환경을 제공할 수 있다.

(2) OOUI를 이용한 클라이언트 인터페이스

클라이언트 측은 웹에 기반한 개방형 분산 HW/SW 통합설계 환경에서 제공되는 서비스의 외형과 느낌을 제공한다. OOUI(Object Oriented User Interface)는 아이콘적인 성향이 강하며, 화면상에서 직접 객체를 조작할 수 있는 객체 지향 사용자 인터페이스이다. OOUI는 예측할 수 없는 순서를 갖는 여러 가지 작업을 수행하는 시스템 설계와 같은 작업에 적합한 메타포(Metaphor)이다. 제안한 환경의 클라이언트 화면에서 아이콘은 실행 가능한 응용 프로그램이 아닌 직접 조작할 수 있는 객체를 표현한다. 각 객체를 선택할 때마다 상황에 맞도록 메뉴가 변경되며, 객체를 드래그&드롭 함으로써 다른 객체와 사용 작용, 이동, 조작된다. 즉, 설계한 작업 데이터 객체를 분할 알고리즘을 수행하는 객체 위에 올려놓으면 작업 환경은 자동으로 HW/SW 분할을 수행하기에 적절한 메뉴로 변경되며 작업 데이터를 이용한 HW/SW 분할이 시각적으로 수행되는 것이다. 웹 페이지에 동적인 객체 조작은 자바 빈(Bean)을 이용해 구현한다.

(3) 트랜잭션 관리를 이용한 효율적인 설계 데이터의 전송

3층 클라이언트/서버 환경과 OOUI 개념의 도입은 네트워크의 과도한 통신을 유발할 수 있다. 제안한 환경은 많은 트랜잭션 발생이 예상되는 분산 환경이기 때문에 트랜잭션 관리는 필수적이며, 시스템 설계 데이터는 트랜잭션이 가지는 ACID 속성이 필요한 응용 분야이다. 네트워크 오버헤드를 줄이고, 문제 발생시 일관성 있는 복구를 위해 트랜잭션 처리를 담당할 트랜잭션 처리 모니터(TP Monitor)를 채택한다. 제안한 환경에서 트랜잭션 처리 모니터가 담당하게 될 부분은 프로세스 관리, 트랜잭션 관리, 클라이언트/서버 통신 관리이다.

- 프로세스 관리 : 서버 프로세스 시작, 이들 프로세스로 작업 집중, 실행 감시, 작업량의 균형 유지
- 트랜잭션 관리 : 트랜잭션 처리 모니터 하에서 실행되는 모든 프로그램에 대해 ACID 속성을 보장
- 클라이언트/서버 통신 관리 : 클라이언트와 서비스가 요청-응답, 대화, 큐잉, 게시-가입, 방송 등의 다양한 방법으로 응용 프로그램 요소를 호출할 수 있게 한다.

제안한 환경은 CORBA를 기반으로 하는 분산 환경이므로 ORB를 사용하는 TP Monitor의 변형인 OTM(Object Transaction Monitor)이 필요하다. 현재 ORB 기반의 TP Monitor로는 IBM의 Component Broker와 BEA System의 M3가 있다. 세계 최초의 OTM 제품인 Microsoft의 MTS는 OLE 트랜잭션에 기초한 COM+ 모델만을 지원하므로 적합하지 않다.

3.3 보안 관리

분산 시스템에서 호스트들 사이에 분산된 여러 가지 자원들은 서버에 의하여 제공되는 네트워크 서비스의 형태로 공유된다. 따라서 분산 시스템 환경에서는 소프트웨어 요소간의 상호 작용이 많아지고 캡슐화에 의한 상호 간의 세부적인 내용을 알 수 없으므로 신뢰 범위가 복잡해진다. 웹에 기반한 개방형 분산 HW/SW 통합설계 환경에서 보안을 위협하는 요소들로는 인가받지 않은 설계자의 설계 데이터로의 접근, 합법적인 설계자로의 위장, 통신망 상의 자료의 불법적 도청, 객체들 사이에 전송되는 자료의 변조 등이 있다. 이와 같은 보안 위협 요소를 방지하기 위해 2번째 계층에 보안 서버를 두어 인증(Authentication) 및 접근 통제(Access Control)와 관련된 보안 사항을 제어하도록 하였다.

분산 환경에서 적용 가능한 인증 기술은 커버로스(커버로스), SESAME(Secure European System for Application in a Multivendor Environment), X. 509 등이 있다. 커버로스는 중앙 집중식 인증 서버를 제공하지만, 대칭키 방식에 의존하는 단점이 있다.

X.509는 디렉토리 인증의 표준으로 공개키 암호화 기법과 디지털 서명을 바탕으로 하는 기술이다. 커버로스와 X.509의 장점을 취합한 SESAME는 보안이 보장된 개방형 분산 시스템을 구축하기 위해 응용 계층에 중점을 둔 보안 시스템으로 인증, 접근 제어, 데이터 무결성, 데이터 기밀성, 부인 봉쇄 등을 지원한다. SESAME는 클라이언트와 서버 사이에 전달되는 보안관련 제어 데이터에 대해서만 암호화하도록 하였으며, 사용자 데이터에 대해서는 선택적으로 암호화가 가능하다. 또한 커버로스에서 제시한 인증 기법을 확장하여 사용하며, 분산 환경에서의 표준 보안 인터페이스인 GSS-API를 제공한다. SESAME에서는 데이터를 보호하기 위한 기법으로 커버로스 키 분배 규약을 이용하며, 공개키 암호화 방식도 사용할 수 있도록 하였다. 또한 X.509의 디렉토리 사용자 신임장을 지원한다.

이런 특징 때문에 웹-기반 HW/SW 통합설계 환경의 보안 서버는 RSA 알고리즘을 추가한 SESAME 기술을 사용하여 보안 기능을 수행한다. 보안 서버는 인증 외에도 접근 통제, 비밀 보장, 데이터 무결성을 보장하는 기능이 있다. 설계 데이터를 작성한 설계자가 자신이 작업한 사실을 부인하지 못하도록 하는 부인 방지(Non-repudiation) 기능과 객체의 호출 성공/실패 여부 등에 대한 감사(Adult) 기능은 SESAME에서 기본적으로 제공하는 서비스를 이용한다.

4. WebCEDA

제안한 웹-기반 개방형 분산 HW/SW 통합설계 환경을 위한 프로토타입으로 WebCEDA(Web-based Codesign Environment with Distributed Architecture)를 구현하였다.

4.1 구현 환경

클라이언트를 위한 하드웨어 플랫폼으로는 Windows98을 운영체제로 하는 Pentium II PC를 사용하였다. 시스템 설계자는 Internet Explorer 4 이상, Netscape 4 이상의 웹-브라우저만 있으면 WebCEDA를 사용할 수 있다. 사용자 인터페이스는 HTML 4.0과 Flash 4.0, JavaScript , Java 2를 이용하여 구현했다. HW/SW 통합설계 서버는 Solaris 2.5를 운영체제로 하는 SUN 워크스테이션을 사용하였으며, 웹-서버로는 Netscape Enterprise Server 3.0을 이용하였다. 응용 서버의 구현에 Java 2가 사용되었으며, 리모트 서버는 Solaris 2.7을 운영체제로 사용하는 Sun Ultra 5(HTML 저장소)와 Solaris 2.6을 운영체제로 하는 Sun Enterprise 3000(DB 서버, 응용 프로그램), Redhat Linux 6을 운영체제로 하는 Pentium PC(파일 시스템)를 사용하였다. 데이터베이스 서버는 Oracle 7을 사용하였으며, Synopsys, SpecCharts 변환기[17], CDFG 툴킷[15], HW/SW 통합시뮬레이션 시스템[16], 실시간 시스템을 위한 확장된 Greedy 알고리즘[19]을 사용하는 분할기 등의 응용 프로그램이 사용되었다.

4.2 WebCEDA의 구성과 설계 절차

그림 2는 WebCEDA의 전체적인 구성을 보여준다.

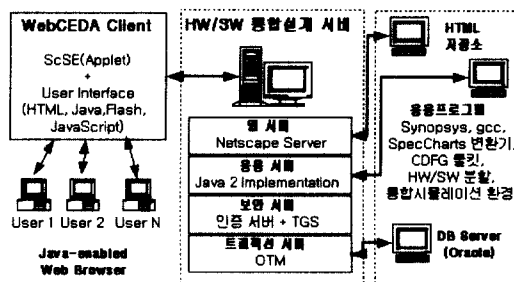



그림 2. WebCEDA 구성도

웹 기반의 HW/SW 통합설계 환경에서 사용자 인터페이스는 일반적인 HTML 문서와 JAVA 애플릿으로 구성한다. HTML이 제공하고 있는 기능만으로는 설계자에게 친숙한 인터페이스를 제공하기 어렵기 때문에, 전체 인터페이스는 Dynamic HTML, Flash 등을 이용해서 효과적인 사용자 인터페이스가 될 수 있도록 동적 웹페이지 형태로 구성하였다. Java를 이용하면 인터페이스는 효과적이 되지만 속도에 문제가 있으므로, 대부분의 작업은 클라이언트에 처리가 넘겨진 HTML과 같은 방법으로 해결하고, 파형 표시나, 도식 편집기와 같은 상호 작용이 많은 그래픽 작업에서만 Java 애플릿을 사용했다. 사용자 인증은 커버로스로 Perl/CGI를 이용해서 구현하였다. 기본적인 통신 프로토콜은 웹 환경의 표준인

HTTP(Hyper-Text Transport Protocol)를 사용하지만, 설계 데이타의 교환은 소켓 프로그래밍을 이용해 구현하였다.

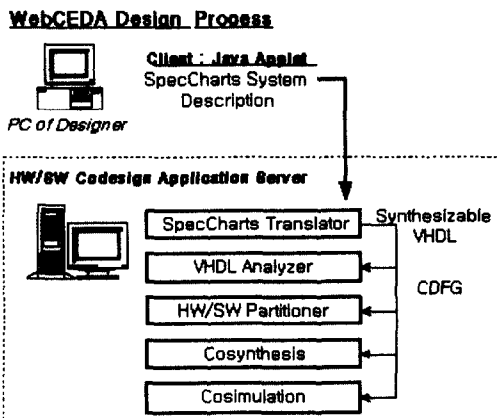
WebCEDA를 이용한 HW/SW 통합설계 절차는 그림 3에서 나타내었다. WebCEDA를 이용한 HW/SW 통합설계는 다음과 같은 순서로 이루어진다.

① 웹-기반 사용자 인터페이스에서 로그인 절차를 통해서 보안 서버와 통신함으로써, WebCEDA의 사용 권한을 얻는다. ② 원하는 시스템을 자바 애플릿 형태의 사용자 인터페이스[18] 내에서 SpecCharts 언어를 통해 기술한다. ③ 자바 애플릿이 가지는 보안 제약을 극복하기 위한 파일 처리 기능을 가지는 HW/SW 통합설계 응용 서버는 시스템 설계자의 저장하기, 불러오기, 명령 실행 등의 요구를 처리한다. 애플릿과 HW/SW 통합설계 응용 서버 사이의 통신은 효율성과 애플릿 보안 제약을 극복하기 위해 소켓 인터페이스를 통해 구현하였다. 그러므로 HW/SW 통합설계 응용 서버의 내부 설계가 RPC, CORBA 등으로 변경되더라도 동일한 인터페이스를 제공할 수 있다. ④ SpecCharts의 그래픽 표현은 SpecCharts 에디터 내에 내장된 변환기를 통해 동일한 의미를 갖는 텍스트 표현으로 바뀌며, 그것은 다시 SpecCharts 변환기를 통해 Synopsys에서 합성 가능한 VHDL 기술로 변환된다. Synopsys의 vhdlan과 vhdldbx를 이용해서 생성된 VHDL에 대한 논리적 오류 검증 및 시뮬레이션을 수행한다. ⑤ SpecCharts 변환기에 의해 생성된 VHDL은 적절한 검증 절차를 마치고 나면, VDT의 VHDL Analyzer[10]를 이용해서 HW/SW 자동 분할을 위한 내부 모델인 CDFG(Control Data Flow Graph)로 바뀐다. ⑥ HW/SW 분할 알고리즘은 CDFG를 하드웨어 요소와 소프트웨어 요소로 분할하며 인터페이스 기능을 추가하여 다시 VHDL과 C로 변환된다. ⑦ 하드웨어와 소프트웨어로 나뉘어진 시스템은 GNU C Compiler와 Synopsys를 통해 합성되며, 통합 시뮬레이션 시스템을 이용하여 성능 평가 및 전체 시스템의 기능 검증을 수행함으로써 완성된다.

그림 3. WebCEDA를 이용한 통합설계 절차

4.3 WebCEDA의 클라이언트

WebCEDA의 사용자 인터페이스는 HTML, CGI, Java, Flash 등의 기술이 혼합된 동적 웹 페이지이다. 시스템 설계자는 WebCEDA의 URL을 웹 브라우저에 입력함으로써 HW/SW 통합설계를 이용한 시스템 설계 작업을 수행할 수 있다. 그림 4는 WebCEDA의 사용자 인터페이스를 보여준다.

그림 4. WebCEDA의 사용자 인터페이스

WebCEDA의 시스템 기술 언어인 SpecCharts는 VHDL과 계층적이며 병렬적 기능을 갖는 상태 다이어그램을 결합했다고 요약할 수 있다. 그럼으로써, VHDL이 가지지 못한 상태전환, 계층성, 예외 처리 등의 기능이 새롭게 추가, 확장되어 내장형 시스템의 설계에 강점을 가진다. SpecCharts를 웹-기반 HW/SW 통합설계 환경의 시스템 기술 언어로 사용하기 위해서는 SpecCharts 언어를 합성할 수 있는 합성툴을 제작해야 한다. 그러나 SpecCharts 언어를 위한 전용 합성툴을 설계하는데는 많은 비용이 들어가기 때문에, SpecCharts를 이용해 시스템 기술이 가능한 에디터와, 같은 의미를 가지는 VHDL 언어로 변환시

켜 주는 변환기를 설계해서 기존의 VHDL 합성툴들을 이용하는 것이 더 효과적이다. 현재 SpecCharts를 지원하는 SpecSyn과 같은 통합 설계 관련 툴이 개발 중에 있지만, 플랫폼이 제한적이어서 웹을 기반으로 하는 본 연구에는 적합하지 않다. 이런 이유로 본 연구에서는 SpecCharts의 그래픽 표현을 사용해서 시스템을 기술할 수 있는 자바 애플릿을 개발하여 시스템을 설계할 수 있도록 하였다. 본 논문에서 개발한 ScSE는 SpecCharts의 그래픽 모드와 텍스트 모드의 설계를 모두 지원하며, 기본적으로 그래픽 모드를 우선적으로 이용하도록 되어 있어 시스템 설계자로 하여금 상향식 접근 방식으로 시스템을 설계하도록 유도하였다. 설계된 결과는 SpecCharts 변환기에 의해 VHDL로 변환되며 Synopsys와 같은 기존의 합성툴을 이용해서 시뮬레이션을 수행하고, van(VHDL Analyzer)를 이용해서 CDFG로 변환되어 HW/SW 분할 알고리즘을 적용시킨다. 그림 5는 Java로 구현된 ScSE의 클래스 계층을 CASE 도구인 Plastic을 사용하여 UML로 나타낸 것이다.

그림 5. ScSE의 주요 클래스 계층도

4.4 ScSE

시스템 설계자는 인증 절차를 마치고 나면, 웹-기반 인터페이스의 ScSE 버튼을 클릭 함으로써 본 실험실의 웹 서버에 구현되어 있는 ScSE를 다운로드 받는다. ScSE는 기본적으로 SpecCharts의 그래픽 표현을 사용해서 시스템을 기술하도록 되어 있다. ScSE에서 기술한 그래픽 표현의 SpecCharts 기술은 내장된 변환기를 통해 텍스트 표현으로 변경되며, WebCEDA의 서버 환경에 포함된 응용 프로그램 서버와의 소켓 통신을 통해서 서버에 저장된다. ScSE의 메뉴에서 VHDL 변환 기능을 선택하면, 응용 프로그램 서버는 저장된 SpecCharts 텍스트 표현을 SpecCharts 변환기를 이용해서 VHDL로 변환시켜서 그 결과를 돌려준다.

SpecCharts를 이용한 시스템 설계 과정은 전체적인 시스템을 기본 객체인 Behavior를 계층적으로 구성함으로써 시작한다. 계층적 구조를 갖는 각각의 Behavior들은 내부적으로 또 다른 Behavior 또는 VHDL 문장을 포함하며, 전이 아크(Transition Arc)를 통해서 상태 전환이 이루어진다. Behavior는 순차적, 혹은 병렬적으로 수행되는 하위 상태를 포함하거나 그 자체로써 하나의 VHDL 문장이 되는데, 각각은 Concurrent Subbehavior, Sequential Subbehavior, Code로 표현된다. 전이 아크는 TOC(Transaction-On-Complete ion)와 TI(Transaction-Immediately) 2종류이다. 각각은 <T, C, NB>의 형식으로 표현되는데, T는 전이의 종류를 나타내며, C는 전이가 발생하는 상태를 뜻한다. NB는 Next Behavior의 약자로 전이의 발생으로 인해 어떤 Behavior로 상태가 전이될 것인지를 나타낸다. SpecCharts에서의 동기화는 VHDL에서와 같이 wait 문을 이용해서 이벤트의 발생을 검사함으로써 이루어지며, TI 아크를 이용함으로써 예외상황에 대한 처리를 할 수 있다. SpecCharts는 그래픽 표현(Graphical Representation)과 텍스트 표현(Textual Representation)이라는 2가지 모드로 시스템을 기술한다.

ScSE의 구현에 사용된 기술은 크게 객체 직렬화(Object Serialization), 소켓 프로그래밍, 스윙 인터페이스의 3가지로 요약된다.

(1) 객체 직렬화를 이용한 지속적 객체 저장 기술 : 객체 직렬화는 실행시간 객체를 위한 저장 방식으로 플랫폼에 무관하게 설계 자료를 저장할 수 있다. 이 기능은 간단한 인터페이스 상속만으로 기능 구현이 가능한데, 원격지의 저장 객체를 활용할 수 있는 지속적 저장 메커니즘을 제공한다. 저장된 객체는 원격지간에도 다른 실행시간에 저장된 객체를 불러와 완전히 복구 가능하다.

(2) 소켓을 이용한 애플릿 상에서 클라이언트-서버 통신 : 기존 브라우저들은 보안상의 문제로 인해 서버측 접근을 제한해두었기 때문에, 애플릿 상에서의 저장 기능 구현이 불가능했다. 이를 극복하고, 로컬 저장을 넘어선 인터넷상에서 원격지 팀 프로젝트 환경을 지원하기 위해 소켓 프로그래밍 기법을 사용하였다. 클라이언트가 브라우저를 이용하여 애플릿

을 실행할 때 애플릿은 내부적으로 소켓을 생성하여 서버와 연결된다. 서버와 클라이언트는 소켓을 이용하여 통신하면서 내부 프로토콜에 따라 설계 데이터를 서버에 저장한다. 현재 이 기능은 단순한 내부 프로토콜로 구현되어 보안에 취약한 약점이 있으며 이는 차후에 보완할 예정이다.

(3) 스윙을 이용한 인터페이스 구성 : 자바의 AWT를 이용한 기존 인터페이스는 시스템 설계 작업을 수행하기에 충분한 인터페이스를 제공하기 힘들었다. 스윙은 JFC의 일부인 광범위한 인터페이스 구성 라이브러리이다. ScSE는 스윙을 이용해 직관적이며 세련된 인터페이스를 구현하였으며, 플랫폼에 종속되지 않는 안정된 룩 앤 필(Look and Feel)을 지원하였다.

ScSE는 비교적 쉬운 사용자 인터페이스로 구성되었으며, 디스플레이의 한계를 극복하기 위해 하위 Behavior로 설계를 확장하는 다이브(Dive) 기능, SpecCharts의 그래픽 표현을 텍스트 표현으로 바꾸는 기능, HW/SW 통합설계 응용 서버와의 네트웍 연결 및 신뢰성 있는 파일 전송 기능 등을 포함한다. 그림 6은 업/다운 카운터를 ScSE를 통해 기술한 모습을 보여준다.

그림 6. ScSE의 인터페이스

각 behavior는 그 속성에 따라 서로 다른 색으로 표현되어 설계자는 직관적으로 어떠한 behavior인지 파악할 수 있다. Concurrent subbehavior들의 경계에 표시된 점선은 병렬성을 나타낸다. TI 아크는 화살표, TOC 아크는 끝에 종료점이 달려 있는 화살표로 표시된다. 각 전이 아크는 전이가 발생되는 이벤트를 표시함으로써 전체 명세를 쉽게 파악할 수 있도록 하였다. Sequential Subbehavior 중 시작이 되는 behavior는 역삼각형으로 표시되어 있다. SpecCharts의 그래픽 표현은 전체 Behavior의 계층성, 예외 처리, 상태 전환을 체계적으로 보여주지만, 프로그래밍 코드를 직접 각 Behavior에 나타내게 되면 전체 화면이 오히려 복잡해진다. 이 문제를 해결하기 위해, 각 Behavior나 전이 아크를 더블 클릭 하거나 좌측의 툴바에 있는 아이콘을 이용해 코드 윈도우를 띄우는 방법을 채택하였다. 입력된 각 Behavior나 전이 아크는 속성 윈도우를 통해 크기나 위치 정보를 변경할 수 있으며, 마우스를 이용해서 객체를 드래그 앤 드롭해서 조작할 수도 있다. 각 전이 아크는 그 이름과 전이가 발생할 조건을 기술하도록 하였으며, 전이 아크의 종류와 전이의 출발점과 도착이 되는 Behavior는 자동으로 입력되도록 하였다.

4.5 HW/SW 통합설계 서버

HW/SW 통합설계 서버는 웹 서버, 응용프로그램 서버, 트랜잭션 서버, 보안 서버로 구성된다. 웹 서버와 트랜잭션 서버는 상용 제품을 그대로 사용하기 때문에 응용프로그램 서버와 보안 서버에 대해서만 설명한다.

응용프로그램 서버는 ScSE에서 설계된 자료를 서버에 저장하고, 설계자의 저장된 자료에 대한 요청을 처리하며, 리모트 서버에 있는 CDFG 툴, Synopsys, HW/SW 통합설계 환경과 같은 관련 응용 프로그램을 실행하고, 그 결과를 브라우저로 보내는 역할을 수행한다. 그림 7은 응용프로그램 서버의 클래스 계층도를 CASE 도구인 Plastic을 이용하여 분석한 다음 UML로 나타낸 것이다.

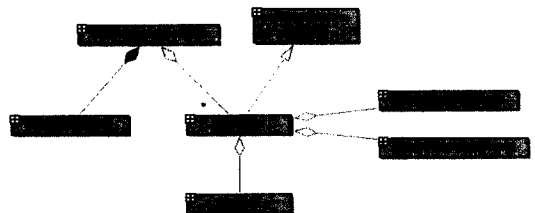



그림 7. 응용프로그램 서버의 주요 클래스 계층도

응용프로그램 서버의 동작 과정은 다음과 같다.

서버는 포트 번호를 이용해서 서버 소켓을 생성하고, 포트에 접근하는 클라이언트가 있을 때까지 대기한다. 매번 포트에 대한 접근 응답이 있을 경우, 클라이언트 핸들러를 생성한다. 핸들러는 클라이언트 소켓을 생성하며, 클라이언트 소켓으로부터 입출력 스트림을 생성한다. 클라이언트 소켓으로부터 메시지를 수신하고, 수신된 메시지를 통해 시스템 설계자의 명령을 판별한다. 명령이 쓰기인 경우, 다음 소켓으로부터 수신된 내용을 서버 측에 저장한다. 읽기의 경우, 서버 측에 쓰여진 내용을 소켓으로 전송하고, 실행인 경우에는 서버에 있는 명령을 실행하고 그 결과를 클라이언트에게 돌려준다.

WebCEDA의 보안 서버는 커버로스와 SSL을 이용하여 구현하였다. 웹 브라우저간의 암호화 통신을 위해서 SSL을 지원하는 Netscape사의 웹 브라우저와 서버를 이용하였다. 그림 8은 보안 서버의 구성을 보여준다.

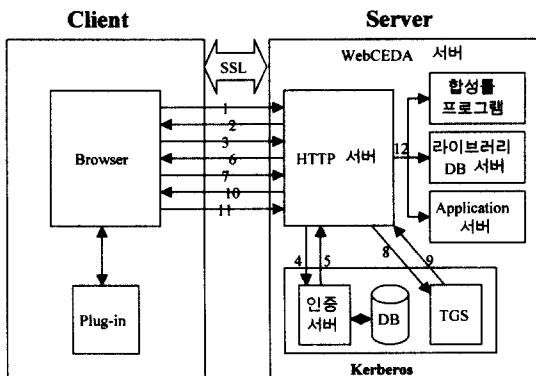



그림 8. WebCEDA의 보안 서버 모델

클라이언트와 서버측의 커버로스 암호 프로토콜을 이용한 사용자 인증 방식에 대한 동작 과정은 다음과 같다.

(1) WebCEDA 환경을 이용하기 위해서 웹 브라우저가 웹 서버에게 서비스 요청

(2) 웹 서버는 웹 브라우저에게 플러그-인을 실행

(3) 웹 브라우저는 플러그-인을 이용하여 사용자 정보(ID와 패스워드)와 TGS(Ticket Granting Server)에 대한 접속을 요구하는 메시지를 전송

(4) 웹 서버는 CGI 프로그램을 이용하여 전송 받은 사용자 정보(ID와 패스워드)와 TGS에 대한 접속 요구 메시지를 커버로스 인증 서버에게 전송

(5) 커버로스 인증 서버는 데이터베이스에 들어있는 사용자의 접근 권한을 검증하고, 웹 브라우저와 TGS간의 세션키를 사용자의 정보인 패스워드로 암호화하고, 사용자와 TGS간의 TGT(Ticket Granting Ticket)를 인증서버와 TGS간의 비밀키로 암호화시켜 전송

(6) 웹 서버는 사용자의 패스워드로 암호화된 세션키와 TGT를 웹 브라우저에게 전송

(7) 웹 브라우저의 플러그-인은 암호화된 티켓인 TGT와 세션키를 이전에 사용자가 입력한 패스워드를 이용하여 복호화를 수행한다. 그 결과 사용자와 TGS간의 세션키는 알 수 있지만, TGS의 비밀키로 암호화된 TGT의 정보는 알 수 없다. 플러그-인은 TGT와 복호화시킨 세션키로 사용자 인증정보를 암호화한 후 웹 서버에게 전송.

(8) 웹 서버는 CGI를 이용하여 TGS에게 전송받은 TGT와 사용자 인증정보를 전송

(9) TGS는 인증 서버간의 비밀키로 TGT를 복호화하고 복호화한 내용 중에서 사용자와 TGS간의 비밀키로 사용자 인증정보를 다시 복호화한 후 두 내용을 비교 검증 검증후 일치하면 SGT(Service Granting Ticket)을 해당 서비스 서버와 TGS간의 공유된 비밀키로 암호화하고 사용자와 해당 서비스 서버와의 세션키를 사용자와 TGS간의 세션키로 암호화하여 웹 서버에게 전송

(10) 웹 서버는 브라우저의 플러그-인에게 전송

(11) 브라우저의 플러그-인은 전송받은 암호화된 메시지를 사용자와 TGS간의 세션키로 복호화를 수행한다. (7)번에서와 같이 SGT는 사용자가 서비스 서버의 비밀키를 알지 못하기 때문에 복호화되지 않는다. 그러나, 사용자와 서비스 서버와의 세션키는 알 수 있다. 이렇게 얻은 세션키로 사용자 인증 정보와 SGT를 웹 서버에게 전송한다.

(12) 웹 서버는 CGI를 이용하여 해당서버에 전송한다. 그리고, 해당 서버는 TGS와 공유된 비밀키로 SGT를 복호화시키고, 그 결과로 사용자와 서버간의 세션키를 얻어서 사용자 인증정보 내용을 다시 복호화시킨후, 두 내용을 비교 검증한다. 일치하면 향후 해당 서버와 사용자사이의 메시지를 암호화하는데 사용하거나 혹은 새로운 랜덤 세션키를 교환하는데 사용된다.

WebCEDA에는 웹 서버를 비롯한 커버로스의 인증 서버와 TGS가 포함된 보안 서버, 라이브러리 데이터 베이스 서버, 응용프로그램 서버, 트랜잭션 서버들이 있다. 이런 각 서버들은 웹 서버를 통해서 웹 브라우저와 통신을 해야 하기 때문에 웹 서버의 기능이 반드시 확장될 필요가 있다. 따라서, 이런 모듈을 만들기 위해서 Netscape의 NSAPI를 이용해 웹 서버가 Access Handler, Authentication Handler, Response Handler의 기능을 갖도록 확장하여 구현하였다. NSAPI는 다양한 변수들을 추가시킬 수 있도록 제공하고 있으며, 어떤 모듈에서나 접근할 수 있는 데이터 구조를 가진다. 웹 브라우저의 사용자 정보를 커버로스 인증 서버에 저장하기 위해서 Oracle 데이터베이스 서버를 이용하여 데이터베이스로 구축하였다. 사용자는 웹 브라우저로 사용자 인증을 거친 후 WebCEDA를 이용할 수 있다. WebCEDA의 보안 서버가 제대로 동작하기 위해서는 기존의 웹 브라우저의 기능 역시 확장해야 한다. 본 논문에서는 브라우저의 확장된 기능을 위해서 Netscape의 플러그-인 기법을 이용해서 커버로스를 지원하기 위한 플러그-인을 구현했으며 기능은 다음과 같다.

- 커버로스 인증 서버에서 전송된 암호화된 티켓인 TGS 및 SGT의 정보를 저장시키고, 해당 세션키를 얻기 위해서 복호화 가능
- 사용자 인증정보(Authentication)을 생성

커버로스의 구현은 공개된 Bones를 사용하였다. 커버로스는 미국의 수·출입제한 정책으로 인해서 암호화 루틴을 호출하는 부분이 없기 때문에 실험을 위해서 DES 암호 알고리즘을 실제로 Bones의 부분에 삽입해서 실험을 하였다. 웹 브라우저의 플러그-인은 Visual C++ 6.0을 이용하여 구현하였다.

시스템 설계자가 WebCEDA의 사용자 등록 화면 절차를 마치고, 초기화면에서 로그-인 버튼을 클릭하면 자동적으로 웹 서버에 의해서 브라우저의 플러그-인이 실행된다. 설계자가 자신의 ID와 패스워드를 입력하고 클릭하면 이 정보가 SSL로 암호화되어 보안 서버로 전송된다. 커버로스 인증 서버에서 이 사용자 정보를 데이터베이스에 등록된 정보와 비교하여 사용자에게 티켓과 세션 키를 전송하고, 플러그-인은 다시 이를 암호화 및 복호화를 거쳐 해당 서버로의 접근을 시도한다. 정상적인 과정을 거치고 나면, 화면 하단에 WebCEDA의 기능을 이용할 수 있는 버튼이 생긴다. 이 후로는 계속 해당 서버를 사용할 수 있다. 다른 응용서버를 사용하려면 TGS의 티켓과 해당 응용서버의 세션 키만으로 곧바로 접근 권한이 생긴다. 물론, 티켓에 포함된 티켓의 유효시간이 지나면 다시 티켓을 요구해야 한다.

4.6 리모트 서버

WebCEDA의 리모트 서버는 SpecCharts 변환기, Synopsys, gcc 등의 툴과 데이터베이스(Oracle 7), 파일 시스템, HTML 저장소 등으로 구성된다. 이 중에서 본 연구에서 개발한 SpecCharts 변환기와 HW/SW 분할 알고리즘에 대해서만 설명한다.

SpecCharts 변환기는 SpecCharts로 기술된 시스템 스펙을 HW/SW 통합설계의 전반적인 절차를 통해 사용할 수 있도록 하기 위해서는 SpecCharts를 같은 의미를 가지는 VHDL으로 변환시키는 역할을 수행한다. SpecCharts에서 문법적으로 VHDL과 비교되는 점은 Behavior 문과 전이 아크가 추가되었다는 점이다. 전체적인 SpecCharts 기술은 크게 Entity와 Architecture로 나뉘는데, Entity 선언은 VHDL과 같다. Architecture 선언 내부만이 전이 아크로 묶여진 계층화된 Behavior로 이루어진다는 점에서 VHDL과 차이가 있을 뿐이다. 따라서, SpecCharts 변환기는 범위를 축소시켜 SpecCharts내의 Behavior 문과 전이 아크를 통한 상태전환, 계층성, 예외 처리 기능을 동일한 기능의 VHDL 코드로 바꾸어주는 일을 한다고 말할 수 있다. SpecCharts의 Behavior 중에서 Code는 그 자체가 VHDL 문장으로써, 특별한 변환 과정을 거치지 않는다. SpecCharts 변환기 설계시 가장 먼저 고려해야할 점은 복합된 Behavior를 VHDL로 표현하는 방법이다. SpecCharts의 계층성을 VHDL 상에서 표현할 수 있는 유일한 방법은 중첩된 Block문을 사용하는 것이다. 전체적인 알고리즘은 하나의 Behavior를 식별할 때마다 다시 변환 알고리즘을 적용하는 재귀적 방법에 의해 구성되어 있다. SpecCharts 변환기의 알고리즘은 하위 Behavior가 서로 병렬적으로 수행되어야 한다면, process문을 통해 그 기능을 VHDL로 구현할 수 있다. 또한 상태 전환 기능을 VHDL 을 통해 수행되도록 하기 위해서 각 Behavior를 활성화하는 신호와 그 Behavior의 연산이 끝났음을 알리는 신호가 생성되어야 한다.

이 과정이 끝나면, Behavior내의 VHDL 코드를 그대로 출력 파일에 추가한다. 전이 아크는 loop문 속에서 앞서 생성한 신호를 wait문을 통해 처리하도록 함으로써 구현 가능하다. 만일 TI 아크가 포함되어 있는 경우, 예외 처리를 위해서 폴링 기법을 적용시켜 인터럽트 신호를 잡아내도록 한다. 모든 과정이 끝나면, 그것을 다른 Behavior에게 알리는 신호를 발생시키고 블럭을 닫으면, 하나의 Behavior가 VHDL로 변환된다. 각 과정은 재귀적으로 수행되므로 모든 Behavior가 VHDL로 변환되고 나면 SpecCharts로 기술된 전체 시스템은 동일한 기능의 VHDL로 표현되어 진다. SpecCharts의 구현에는 Flex와 Bison이 사용되었다.

WebCEDA에서 사용하는 HW/SW 분할 알고리즘은 비용-효과적 실시간 시스템의 설계를 위해 확장된 Greedy 알고리즘이다. 그림 9는 WebCEDA에서 사용하는 확장 Greedy 알고리즘을 보여준다.

```
ExtendedGreedy(CDFG)

P= {0,∅}  /*all-hardwareinitialpartition*/
CDFG = periodCM(CDFG)
schedulability = priorityCM(CDFG)
while schedulability == true
    min_cost = calculateCost(CDFG)
    max_cost = ∞
    repeat
        P_orig = P
        cost = cur_cost = min_cost
        for each o_i ∈ hardware loop
            AttemptMove(P,o_i)
        end loop
    until min_cost > cur_cost
end while

procedure AttemptMove(P,o_i)
if SatisfiesPerformance(Move(P,o_i)) and
    Objfct(Move(P,o_i)) < Objfct(P) then
    P = Move(P,o_i)
    cost = cost - calculateCost(CDFG)
    for each o_j ∈ Successors(o_i) loop
        AttemptMove(P,o_i)
    end loop
end if
```

그림 9. 확장된 Greedy 알고리즘

알고리즘에서 정확한 결과 산출을 위한 자원 할당(Allocation)과, 각 평가 항목의 추정(Estimation)에 관한 항목은 생략되었다. 확장된 Greedy 알고리즘의 입력은 SpecCharts 변환기에 의해 생성된 VHDL을 VHDL 분석기에 입력해서 만들어진 CDFG이다. WebCEDA의 내부 모델로 사용하고 있는 CDFG는 현재 HW/SW 분할 알고리즘에서 가장 많이 사용되고 있는 내부 모델로써, 특히 실시간 시스템의 설계에 강점을 지닌다. 그림 9의 알고리즘에서 periodCM 함수는 CDFG에 필요한 시간 정보를 추가하고, priorityCM 함수에 의해 스케줄 가능성이 테스트되고 있음을 알 수 있다. HW/SW 통합설계 기법을 실시간 시스템의 설계에 적용하기 위해서는 기존의 HW/SW 분할 알고리즘에서 사용하는 일반적인 성능/가격 제약 조건 외에도 시간 제약 조건, 우선 순위 또는 선행 관계 제약 조건, 테스크간 통신 및 동기화 제약 조건 등이 고려되어야 한다. 실시간 시스템을 위한 HW/SW 분할 알고리즘은 성능과 가격이 아무리 좋다고 하더라도 시간 제약 조건을 만족하지 못하는 결과를 산출하지 않음을 보장해야만 하기 때문에, CDFG는 각 노드에 시간 정보를 포함할 수 있도록 확장되었다. 주어진 종료 시한(Deadline)으로부터 각 노드의 시간 정보를 추출해 내기 위한 방법으로 기간 측정 기법(Period Calibration Method)를 사용한다. 또한 각 노드가 수행하는 연산이 종료 시한 내에 이루어지는 것을 보장하는 스케줄 가능성 테스트(Schedulability Test)를 위해 우선순위 상한 프로토콜(Priority Ceiling Protocol)을 사용한다. 우선순위 상한 프로토콜은 세마포어에 의해 보호되는 공용 자원에 배타적으로 접근할 수 있도록 주기적 작업을 스케줄링하는데 사용되는 방법으로 예측 가능성이 높은 장점을 지닌다. 확장 Greedy 알고리즘은 하드웨어 자원 공유를 통한 비용의 감소를 위해 CDFG 모델을 더 이상 나눌 수 없는 기능 객체(Functional Object)로 나누고, 각각의 기능 객체를 연산 단위로 묶는다. 각각의 연산에는 메모리 주소, 버스 등과 같은 실제적인 시스템 요소를 할당(Allocation)된다. 할당이 이루어진 연산은 근접 함수(Closeness Function)와 Greedy 알고리즘을 적용하여 하드웨어와 소프트웨어로 분할되며, 목적 함수(Objective Function)에 의해 분할 결과의 적합성이 평가된다. 목적 함수의 결과가 제약 조건을 위배하면, 제약 조건을 만족할

때까지 확장된 Greedy 알고리즘을 반복적으로 적용한다. 시스템 설계자는 HW/SW 분할을 시작하기 전에 가격, 종료 시간, 통신 오버헤드, 전력 소모량, 칩 면적, 핀 수, 테스트 가능여부, 프로그램 크기, 데이터 크기 등의 제약 조건을 명시할 수 있으며, 이 항목들은 추정(Estimation) 작업을 할 때, 설계가 제대로 되었는지를 평가하는 측정 기준으로도 사용된다. 근접 함수와 목적 함수에 사용될 측정 기준(Metrics)은 6가지로서 가격, 종료 시간, 전력 소비량, 면적, 핀 수, 프로그램 및 데이터의 크기이다. HW/SW 분할 알고리즘의 결과는 하드웨어 요소와 소프트웨어 요소로 분할된 CDFG 형태로 나오게 된다.

5. 결론 및 향후 과제

본 논문에서는 기존의 HW/SW 통합설계 도구가 갖는 플랫폼 제한, 새로운 기술 추가의 어려움, 공동 작업의 불편함 등의 단점을 해결하기 위해, 웹에 기반한 개방형 분산 HW/SW 통합설계 환경을 제안한다. 제안한 환경은 3층 클라이언트/서버 구조를 가지며 세션 서비스를 이용한 협동 작업 환경, OOUI를 이용한 사용자 인터페이스, 트랜잭션 관리, 보안 관리 등의 특징을 통해 기존의 통합설계 도구가 갖는 단점을 해결하였다. 제안한 환경의 타당성을 입증하기 위해 개발한 WebCEDA는 클라이언트, HW/SW 통합설계 서버, 리모트 서버로 구성된다. WebCEDA는 SpecCharts를 시스템 기술 언어로 사용하며, 내부 모델은 CDFG로 표현된다. SpecCharts 변환기를 구현해서 Synopsys, gcc와 같은 기존의 도구를 재사용할 수 있도록 하였다.

본 연구와 관련해서 향후의 연구 과제로는 크게 2가지가 있다. 첫 번째는 WebCEDA에서 사용되는 SpecCharts 변환기나, HW/SW 분할 알고리즘 등을 CORBA 객체로 변환시켜 전체적인 설계 환경이 분산 객체 기술을 이용하여 이루어질 수 있도록 하는 것이다. 현재 객체 래퍼를 이용하여 WebCEDA에서 사용되는 컴포넌트들을 분산 객체화하는 작업이 진행중에 있다. 이런 경우 CORBA/Java의 개선된 보안 기능을 통한 보다 안전한 설계 환경의 구현이 가능하고, 전체 시스템의 유지 보수 비용을 줄일 수 있다. 두 번째는 통합설계 결과를 시각화하는 도구와 전체 설계 절차를 관리하는 도구를 구현하는 것이다. 현재 WebCEDA는 동적 웹 페이지를 통해서 전체 설계 절차를 진행하도록 되어 있어 이 부분을 좀더 편리하게 관리할 수 있는 도구의 개발이 필요하다. 또한 설계 결과가 텍스트 형태로 제공되어 분석이 어렵기 때문에 이를 시각화하는 도구의 개발이 필요하다.

참 고 문 헌

[1] Jerzy Rozenblit, Klaus Buchenrieder, "Codesign : Computer-Aided Software/Hardware Engineering", IEEE PRESS, 1995.

[2] Linda Geppert, "IC Design on the world wide web", IEEE Spectrum, p45-50, April-June, 1998

[3] M. D. Spiller, A. R. Newton, "EDA and the Network", Proc. of the IEEE International Conference on Computer-Aided Design, pp. 470-476, Nov, 1997

[4] S. Narayan, F. Vahid, and D. D. Gajski. "System specification with the SpecCharts language". In IEEE Design & Test of Computers, Dec. 1992.

[5] G. Berry, "A hardware Implementation of pure Esterel", Digital Equipment Paris Research Laboratory, July, 1991.

[6] IOS IS 8807, Information Processing Systems, Open System Interconnection, LOTOS, A Formal Description Technique Based on the Temporal Ordering of Observational Behavior, IOS, July, 1988.

[7] J. Buck, S. Ha, E. A. Lee, and D. G. Messerschmitt, "Ptolemy : A Framework for Simulating and Prototyping Heterogeneous Systems", International Journal of Computer Simulation, Vol 4, pp. 155-182, April, 1994.

[8] R. Esser, J. Teich, L. Thiele, "CodeSign: An Embedded System Design Environment", J. Proc. of the IEEE, Computers and Digital Techniques, pp 171-180, 1998.

[9] F. Balarin, M. Chiodo, P. Giusto, H. Hsieh, A, Jurecska, L. Lavagno, C. Passerone, A. Sangiovanni-Vincentelli, E. Sentovich, K.

Suzuki, B. Tabbara, "Hardware-Software Co-Design of Embedded Systems: The Polis Approach", Kluwer Academic Press, 1997.

[10] R. Niemann, P. Marwedel, "Synthesis of Communicating Controllers for Concurrent Hardware/Software Systems", Design, Automation and Test in Europe (DATE), 1998.

[11] H. Oh and S. Ha, "A Hardware/Software Cosynthesis Technique Based on Heterogeneous Multiprocessor Scheduling," 7th International Workshop on Hardware/Software CO-Design, Rome, Italy, May 1999.

[12] R. Ortega, G. Borriello, "Communication Synthesis for Embedded Systems with Global Considerations", Proc. Codes/CACHE, pp. 69-73, 1997.

[13] D. Gajski, J. Gong, F. Vahid, S. Narayan, "The SpecSyn Design Process and Human Interface," UC Irvine, Technical Report ICS-TR-93-03, 1993.

[14] 신영기, 홍원기, "분산 멀티미디어 협동 작업 환경을 위한 CORBA 기반의 보안성 있는 세션 서비스", 정보과학회논문지(A), 26권, 6호, pp. 670-682, 1999.

[15] http://poppy.snu.ac.kr/VDT, VHDL Developer's Toolkit (VDT)

[16] S. Yoo, K. Choi, "Optimistic Distributed Timed Cosimulation Based on Thread Simulation Model", Proc. Int'l Workshop on Hardware/Software Co-Design(Codes/CASH E), Mar. 1998.

[17] 김승권, 김종훈, "HW/SW 통합설계를 위한 SVT(SpecCharts_to_VHDL Translator)의 설계 기법", 한국정보과학회 가을학술발표논문집, 24권, 2호, pp. 741-744, 1997.

[18] 김승권, 김종훈, "웹-기반 HW/SW 통합설계를 위한 SpecCharts 기술 환경", 한국정보과학회 영남지부 학술발표논문집, 6권, 1호, pp. 157-164, 1998.

[19] 김승권, 박종호, 김종훈, "실시간 시스템의 HW/SW 통합설계를 위한 확장된 Greedy 알고리즘", 한국정보과학회 봄학술발표논문집, 26권, 1호, pp 717-719, 1999.

김 승 권

1997년 동아대학교 컴퓨터공학과 졸업(공학사).
1999년 동아대학교 컴퓨터공학과 졸업(공학석사).
1999년~현재 동아대학교 컴퓨터공학과(박사과정).
관심분야 : HW/SW 통합설계, 보안 프로토콜, 실시간 시스템

김 종 훈

1974년 동아대학교 전자공학과 졸업(공학사).
1977년 동아대학교 전자공학과 졸업(공학석사).
1986년 경북대학교 전자공학과 졸업(공학박사).
1986년~현재 동아대학교 컴퓨터공학과 교수.
관심분야 : HW/SW 통합설계, 실시간 시스템, 정보 보안